Simultaneous optimization of neural network weights and active nodes using metaheuristics

Simultaneous Optimization of Neural Network Weights and Active Nodes using Metaheuristics

Varun Kumar Ojha*, Ajith Abraham*,Václav Snášel*
*IT4Innovations, VŠB Technical University of Ostrava, Ostrava, Czech Republic
varun.kumar.ojha@vsb.cz, ajith.abraham@ieee.org, vaclav.snasel@vsb.cz

***Abstract*—Optimization of neural network (NN) significantly influenced by the transfer function used in its active nodes. It has been observed that the homogeneity in the activation nodes does not provide the best solution. Therefore, the customizable transfer functions whose underlying parameters are subjected to optimization were used to provide heterogeneity to NN. For the experimental purpose, a meta-heuristic framework using a combined genotype representation of connection weights and transfer function parameter was used. The performance of adaptive Logistic, Tangent-hyperbolic, Gaussian and Beta functions were analyzed. In present research work, concise comparisons between different transfer function and between the NN optimization algorithms are presented. The comprehensive analysis of the results obtained over the benchmark dataset suggests that the Artificial Bee Colony with adaptive transfer function provides the best results in terms of classification accuracy over the particle swarm optimization and differential evolution.**

***Index Terms*—Meta-heuristics; Neural network; Activation function; Beta Function; Artificial Bee Colony,**

I. Introduction

Due to the property of being robust and adaptive with the problem environments, the Neural Network (NN) has emerged as the most desirable computational tool for solving nonlinear and complex optimization, pattern recognition, function approximation classification, etc., problems [1], [2]. On the other hand, the meta-heuristic algorithms are well appreciated for their role in the optimization of the Neural Networks (NNs) [3]. The conventional NN optimizations/training algorithms are efficient in local search or in other words, they are efficient in the exploitation of the current solutions for the creation of new solutions. Whereas, the meta-heuristic algorithms are efficient in both exploitation of the current solution and exploration of the given search space for the creation of new solutions. The meta-heuristic algorithms can be used for the optimization of the connection (synaptic) weights, architecture (geometrical arrangement of the nodes), transfer (activation) functions associated with the nodes and the learning mechanisms [1].

Yao [3] has summarized the Evolutionary Algorithm (EA) based optimization of the NN where it can be found that the NN optimization is not only limited to optimization of connection weights, but it encompasses the optimization of network architecture, activation function, and learning rules. In the present research, we have illustrated the meta-heuristic framework for the optimization of NN and investigated the impact of the optimization of the underlying parameters of the transfer function associated with the active nodes (nodes at hidden and output layers) of the neural network. In the past, efforts to optimize the transfer functions were mostly limited to finding the appropriate combinations of different variety of transfer functions at the active nodes of a NN. Liu and Yao [4] have chosen a combination of Sigmoid (Logistic) and Gaussian function to optimize transfer functions of the NN. Similarly, White and Ligomenides [5] opted to combine 80% of Sigmoid and 20% of Gaussian activation function. Castelli and Trentin [6] have illustrated a connectionist model for an adaptive selection of the transfer function. In their model, they selected hidden unit with a pair $\{f, p\}$ where f was a set of various transfer functions and p was the corresponding probabilistic measure of the likelihood of the node that was relevant to the computation of the output over the current input. Alimi [7] and [8] have illustrated the significant benefits of using the Beta function in the optimization of the NNs.

In the present research, we have chosen Logistic, Tangent-hyperbolic, Gaussian and Beta function in the optimization of NN. Unlike the research referred above, in the present research, we were inclined toward the homogeneity in NN active nodes. In the other words, all the active nodes in the NN were set using similar transfer functions. Interestingly, in our experiment we have chosen to optimize the parameters of the transfer function that were set at the active nodes of the NN. Thereby, it imparted heterogeneity between the transfer functions of the NN. A similar approach was adopted by van Wyk and Engelbrecht [9] for the optimization of lambda-gamma NN using particle swarm optimization. We have extended the idea, where we use various transfe function and meta-heuristic algorithms in order to enhance the performance of NN. A meta-heuristic NN optimization framework illustrated in the present paper was used for the simultaneous optimizations of the NN connection weights and the transfer functions. A comprehensive experimental result presented in the paper suggest that setting the active nodes of the NN using the customizable transfer functions whose parameters were optimized using the meta-heuristic algorithms was worth investing efforts. The Artificial Bee Colony (ABC) algorithm used for the simultaneous optimization of the connection weights and the transfer functions was consistent in producing the best result in terms of accuracy in classification of the three classification problem chosen in the present research work.

The rest of the paper is organized as follows: In section II, we discussed the fundamental concept of the NNs, the transfer function and the meta-heuristic algorithms. A discussion on

the experimental design and the meta-heuristic framework for the simultaneous optimization of the NN connection weights and the transfer function parameters is provided in section III followed by the results and discussion in section IV. Finally, a conclusion is provided in section V.

II. Neural Network

Artificial Neural Network (NN) or simply the NN imitates the functioning of human brain basically, the biological nervous system that is a network of the immense interconnections between the vast numbers of biological neurons [1]. The neurons are the smallest processing unit of the nervous system. Similarly, the NN is a network of several processing elements (nodes) which gains its capability of behaving intelligently by meticulous training provided using training examples. The NNs have three basic components, architecture, connection weights and learning-rules. In the present scope of the research, we have chosen feed-forward multilayer NN. Geometrically, the NNs are arranged in layer by layer basis, where, each layer may contain one or more computational nodes. Mathematically, the j^{th} node of a NN may be given as:

$$y_j = \varphi_j \left(\sum_i w_{ji} x_i - b_j \right), \tag{1}$$

where y_j is output of j^{th} node, w_{ij} is connection weight between i^{th} node and j^{th} node, x_i is i^{th} input, b_j is bias at the j^{th} node and $\varphi_j(.)$ is transfer (activation) function at j^{th} node. Since variable x is input (known) and variable y is output (to be computed), we need information of the other remaining variables using a training process that can help to find the optimal values for the variables. Transfer function $\varphi(.)$ is function input x and variables $\{t_1, t_2, \ldots, t_n\}$ where t is a parameter of transfer function $\varphi(.)$. It is worth noticing that variable t is usually kept fixed. In subsequent section, we have discussed various kinds of transfer function that may be used at the nodes of a NN.

A. Transfer Function (TF)

Transfer functions at the active nodes of a NN are used to transfer the net scalar input at an active node to a scalar called activation value or the output value at that node. Consult (1), where y_j on the left hand side is the output value of the j^{th} node evaluated using a transfer function $\varphi(.)$ shown on the right hand side. Basically, the transfer functions limit the net input value at a node to a certain range of values in order to allow a NN to behave in certain ways rather than letting it to behave indiscriminately. Transfer functions may be linear or non-linear. Mostly, the NN is designed to solve non-linearly separable problems. Hence, non-linear transfer functions such as: Logistic (Sigmoid), Tan-hyperbolic, Gaussian, Beta basis function, etc. are be used.

Logistic Function: Logistic function (2) also known as sigmoid function is a unipolar function. The Logistic function (2) has three parameters x, λ and θ, where the variable x indicates net-scalar input value of a node, λ indicates steepness and θ indicates center of the Logistic function. The variables λ and θ control the behavior of Logistic function. The most conventional approach is to keep the variables λ and θ value fixed to one and zero respectively. However, the behavior of the function significantly varied with the variation of the parameters λ and θ. Therefore, approaches to optimize the parameters together with the connection weights of a NN will help in obtaining optimal NN.

$$\varphi(x, \lambda, \theta) = \frac{1}{1 + e^{(-\lambda(x-\theta))}} \tag{2}$$

Tangent-hyperbolic Function: Tan-hyperbolic function defined in (3) receives parameters x, λ and θ that are analogues to Logistic functions. However, unlike the unipolar Logistic function, the Tangent-hyperbolic functions are bipolar functions that produce significantly different impact on the net output of a NN with respect to the one produced by the Logistic functions. The parameter steepness λ and center θ controls the behavior of Tangent-hyperbolic function. Hence, optimum values of these parameters may significantly enhance performance of a NN.

$$\varphi(x, \lambda, \theta) = \frac{e^{(\lambda(x-\theta))} - e^{(-\lambda(x-\theta))}}{e^{(\lambda(x-\theta))} + e^{(-\lambda(x-\theta))}} \tag{3}$$

Gaussian Function: Gaussian function (4) parameterized by variables x, λ and θ, where x is net-input and variables σ and μ are width and mean (center) of the function respectively. Gaussian function is a unipolar function that produces a symmetric shape around center μ. The width and center significantly influence the behavior of Gaussian function. Therefore, optimum values of the parameters will be able to enhance the overall performance of a NN.

$$\varphi(x, \sigma, \mu) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(x-\mu)}{2\sigma^2}} \tag{4}$$

Basis Function: Due to Beta function (7) flexibility and universal approximation characteristics and ability adopt variety different shapes, Alimi [7] used Beta function as an activation function of NN. The Beta function is defined as:

$$\beta(x, x_0, x_1, p, q) = \begin{cases} \left(\frac{x-x_0}{\theta-x_0}\right)^p \left(\frac{x_1-x}{x_1-\theta}\right)^q & \text{if } x \in]x_0, x_1[\\ 0 & \text{Otherwise} \end{cases} \tag{5}$$

where $p > 0, q > 0 x_0, x_1$ are real parameters $x_0 < x_1$ and $\theta = (px_1 + qx_0)/(p+q)$ is center of Beta function. Let $\sigma = x_1 - x_0$ is the width of the Beta function which can be seen as scale factor for distance like $\| x - \theta \|$. Hence, x_0 and x_1 defined as:

$$\begin{aligned} x_0 &= \theta - \tfrac{\sigma p}{p+q} \\ x_1 &= \theta + \tfrac{\sigma q}{p+q} \end{aligned} \tag{6}$$

From (5) and (6) the Beta function is written as

$$\varphi(x, \theta, \sigma, p, q) = \begin{cases} A.B & \text{if } x \in]\theta - \frac{\sigma p}{p+q}, \theta + \frac{\sigma q}{p+q}[\\ 0 & \text{Otherwise} \end{cases} \tag{7}$$

where $A = \left[1 + \frac{(p+q)(x-\theta)}{\sigma p}\right]^p$ and $B = \left[1 - \frac{(p+q)(x-\theta)}{\sigma q}\right]^q$ A detailed discussion on various other types of transfer function

provided by Duch and Jankowski [10] supports our discussion that an activation functions with various shapes and behaviors influence the overall performance of a NN. Therefore, the necessity of optimizing parameters in the optimization of NN is evident. The conventional NN optimization algorithms uses fixed transfer function. Hence, meta-heuristic optimization algorithms provides a robust platform for the optimization of both the connection weights and the transfer function parameters of a NN.

B. Meta-heuristic Algorithms

Meta-heuristic algorithms are stochastic procedures that are efficient in both exploitation of the present solutions and exploration of the given search space. The meta-heuristic algorithms such as: Artificial Bee Colony, Particle Swarm Optimization and Differential Evolution can be used for optimization of both connection weights and transfer function parameter simultaneously.

Artificial Bee Colony (ABC): ABC proposed by Karaboga [11] is a meta-heuristic algorithm inspired by foraging behavior of honey bee swarm. The ABC algorithm uses the population of bees to explore the given search space in order to find the optimal solution for a given problem. The ABC algorithm works as follows: At first a memory of initial food position (candidate solution) is initialized and then the food position is updated by the artificial bees in iterative fashion. The i^{th} candidate in a memory of P solutions, where each of which has M variables can be given as:

$$x_{ij} = x_{ij} + rand(-1,1) \times (x_{ij} - x_{kj}) \tag{8}$$

where, $k \in [1,P]$, and $j \in [i,M]$ and x_{ij} is the comparison between the i^{th}food source and a randomly chosen neighbor k. A food source is abundant if it is not of good quality and hence a new food source is obtained as:

$$x_{i,j} = x_{min,j} + rand(0,1)(x_{max,j} - x_{min,j}), \tag{9}$$

where min and max is the bound of the j^{th} variable. Karaboga and Basturk [12] have illustrated the application of the ABC algorithm for the optimization of NN connection weights that in our experiment was extended to the optimization of both connection weights and the transfer function simultaneously. Similarly, we used Particle Swarm Optimization and Differential Evolution algorithms for the same purpose.

Particle Swarm Optimization (PSO): PSO [13] is a population based meta-heuristic algorithm imitates the mechanisms of the foraging behavior of swarms. The PSO depends on the velocity and position update of a swarm. The velocity in PSO is updated in order to update the position of the particles in a swarm. Therefore, the whole population moves towards an optimal solution. The PSO uses a population of motile candidate particles characterized by their position x_i and velocity v_i inside the $n-$dimensional search space. Each particle remembers the best position (in terms of fitness function) it visited b_i and knows the best position discovered so far by the whole swarm g. At each iteration, the velocity of a particle i is updated according to [14]:

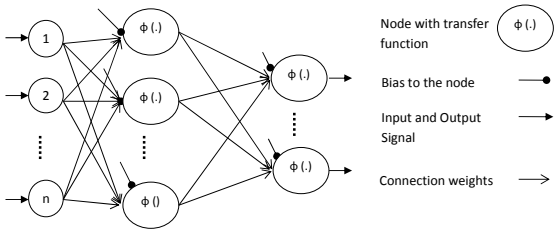

Fig. 1. Phenotype representation of NN

$$v_i^{t+1} = c_0 v_i^t + c_1 r_1^t (b_i - x_i^t) + c_2 r_2^r (\bar{g}^t - x_i^t), \tag{10}$$

where c_1 and c_2 are positive acceleration constants, r_1 and r_2 are vectors of random values sampled from a uniform distribution, vector b_i^t represents the best position known to particle i at iteration t, vector $\bar{y}^t$ is the best position visited by the swarm at time t and inertia factor c_0 is computed as:

$$c_0 = c_0^{max} - \frac{(c_0^{max} - c_0^{min}) \times Current_{iteration}}{Max_{iteration}}. \tag{11}$$

The position of particle i is updated by [14]:

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{12}$$

Differential Evolution (DE): DE proposed by [15] is a popular meta-heuristic algorithm for the optimization of continuous functions. DE has been successfully used for the optimization of NN [16]. The basic principle of DE is as follows: At first an initial population of n dimensional solutions x_i is constructed. The contraction of new solution takes place iteratively. For the purpose of the construction of new solution, three distinct solutions a, b and c are chosen. Thereafter, a random index $N \in [1,n]$ is chosen. Hence, a new solution y_i is constructed as:

$$y_i = \begin{cases} a_i + F \times (b_i - c_i) & \text{if } r_i < \text{CR or } i = N \\ x_i & \text{Otherwise} \end{cases} \tag{13}$$

where CR indicates the crossover rate, F indicates the weight factor and r_i is a uniform random sample chosen in $(0,1)$.

III. Meta-heuristic Framework for Transfer function optimization

Meta-heuristic algorithms have proven their competency in optimizing NNs [3], [17] over the conventional NN optimization algorithms such as Backpropagation [18]. In the present research, we have illustrated the role of meta-heuristic algorithms such as: ABC, PSO and DE (version $DE/rand-to-best/1/bin$ [19]) in the simultaneous optimization of the NN connection weights and transfer function parameters. A three layered feed-forward NN (phenotype) given in Figure 1 represented as a solution vector (genotype) shown in Figure 2 was used for the experiment purpose. It may be noted that, the hidden layer and output layer consist of transfer functions.

Liu and Yao [4], Weingaertner *et al.* [20] and others [21], [22] have adopted heterogeneity in the NN nodes by the

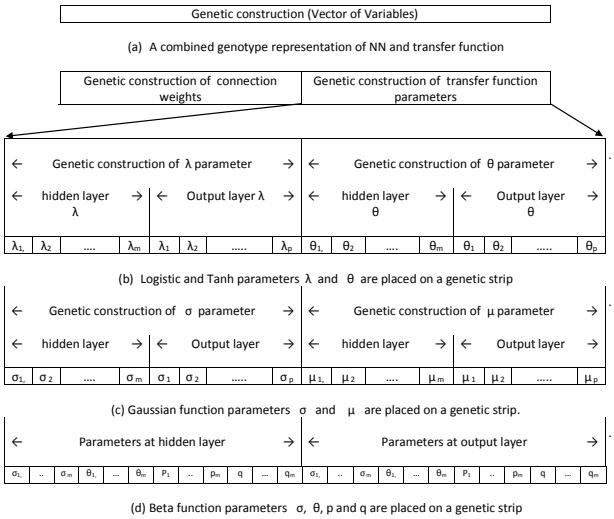

Fig. 2. Genotype representation of NN

procedure META-HEURISTICS-NN(W, ϵ)
  Initialize W_0
  Fittest solution $\mathrm{w}^* = fittest(\mathrm{W}^0)$
  repeat
    $\mathrm{W}^{t+1} := \mathrm{MH}_{\mathrm{Operator}}(\mathrm{W}^t)$
    $\hat{\mathrm{w}} = fittest(\mathrm{W}^{t+1})$
    if $\bar{\mathrm{w}} < \mathrm{w}^*$ **then**
      $\mathrm{w}^* = \bar{\mathrm{w}}$
    end if
  until *Stopping criteria* ϵ *satisfied*
  return w^*
end procedure

Fig. 3. Meta-heuristic Framework for Optimization

means of choosing various transfer function at the hidden layers and the output layers of a NN. On the contrary to their approach, our approach was to explore the impact of the optimization of the individual transfer function parameters on the performance of NN. A meta-heuristic framework for the simultaneous optimization of NN and its transfer function parameters is illustrated in Figure 3, where the meta-heuristic operator $\mathrm{MH}_{\mathrm{Operator}}$ were defined as per the respective meta-heuristic algorithms. For the experiment, the initial population was constructed using the genotype illustrated in Figure 2. We have chosen the genotype representation with Logistic, Tangent-hyperbolic, Gaussian and Beta basis function.

The performance of individual meta-heuristic algorithms are subjected to their respective parameter setting. A list of parameter setting used in the experiment for the ABC, PSO and DE is shown in Table I. Apart from the given parameter, the Mersenne-Twister algorithm with random seeds ware used for the initialization of the initial population within a search space $[-1.5, 1.5]$.

TABLE I
PARAMETER SETTING OF THE ALGORITHMS USED IN THE EXPERIMENTS

Algorithm	Population	Iteration	Other
BP	10	1000	$\eta = 0.5$ and $m = 0.1$
ABC	10	1000	$\mathrm{trial}_{\mathrm{limit}}$ = 100
PSO	10	1000	$c_1 = c_2 = 2.0$, $c_0^{max} = 1.0$ and $c_0^{min} = 0.0$
DE	10	1000	CR = 0.9, F = 0.7

TABLE II
THE 10 CV RESULTS USING BACKPROPAGATION ALGORITHM

Function	Iris		wdbc		Wine	
	Error	Var	Error	Var	Error	Var
SigFix	0.916	0.017	0.933	0.011	0.914	0.069
TanhFix	0.956	0.009	0.945	0.012	0.955	0.017

IV. RESULTS AND DISCUSSION

For the experiment purpose, three benchmark dataset (classification problem) from the UCI Machine Learning repository (http://archive.ics.uci.edu/ml/datasets.html) were used. The dataset chosen were Iris, Breast Cancer (Wdbc) and Wine. In order to justify the significance of the proposed model, the primary independent benchmark results given in Table II in terms of 10 Cross-Validation (CV) over the mentioned dataset was obtained using BP algorithm [18] that has learning rate η and momentum m as its controlling parameters. The experiment using BP was repeated for the Logistic (SigFix) and Tangent-hyperbolic (TanhFix) function for each of the mentioned dataset. In the experiment using BP algorithm, the parameters λ and θ of the transfer functions SigFix and TanhFix ware set to one and zero respectively.

The performance of meta-heuristic algorithms used for the optimization of NN weights and transfer function was tested with the reference to the results obtained using the BP algorithm. Each of the mentioned meta-heuristic algorithms were used optimization of NN independently over the mentioned datasets. The obtained results over Iris, Cancer and Wine classification problems are shown in Tables III, IV and V respectively. The best classification accuracy for the Iris dataset obtained using the BP algorithm with TanhFix was 95.6%. Whereas, the best classification accuracy using the ABC with Beta function over the same dataset was found to be 98.3%. It may also be observed from Table III that optimization of the parameters of the Logistic (SigAdp), Tangent-hyperbolic (TanhAdp), Gaussian and Beta provides classification accuracy better than that of the classification accuracy obtained using the functions with fixed parameter setting. Similarly, the best classification accuracy obtained by the BP over the datasets Cancer and Wine was 94.5% and 95.5% respectively and the best classification accuracy over the same dataset using the meta-heuristic algorithms was 97.0% each. The results obtained over the dataset Iris, Cancer and Wine providing significant evidence that the optimization of functions together with the NN weights helped in obtaining better results than that of using function with fixed parameter setting. Interestingly, the results shown in Tables III, IV and V suggest that the ABC

TABLE III
10CV RESULTS ON IRIS CLASSIFICATION PROBLEM

Function	ABC		PSO		DE	
	Error	Var	Error	Var	Error	Var
SigFix	0.774	0.248	0.799	0.271	0.729	0.190
SigAdp	**0.972**	0.014	0.839	0.669	0.859	0.322
TanhFix	0.943	0.012	0.959	0.013	0.885	0.201
TanhAdp	**0.978**	0.008	0.759	0.150	0.846	0.179
Gaussian	**0.977**	0.020	0.767	0.748	0.893	0.065
Beta	**0.983**	0.008	0.839	0.188	0.944	0.074

TABLE IV
10CV RESULTS ON CANCER CLASSIFICATION PROBLEM

Function	ABC		PSO		DE	
	Error	Var	Error	Var	Error	Var
SigFix	**0.941**	0.006	0.909	0.013	0.928	0.016
SigAdp	**0.958**	0.008	**0.970**	0.016	0.928	0.027
TanhFix	**0.958**	0.007	**0.957**	0.011	**0.951**	0.010
TanhAdp	**0.963**	0.005	0.938	0.009	0.943	0.011
Gaussian	**0.951**	0.005	0.874	0.110	0.906	0.008
Beta	**0.954**	0.012	0.914	0.019	0.912	0.013

TABLE V
10CV RESULTS ON WINE CLASSIFICATION PROBLEM

Function	ABC		PSO		DE	
	Error	Var	Error	Var	Error	Var
SigFix	**0.962**	0.018	0.814	0.188	0.861	0.083
SigAdp	**0.986**	0.019	0.679	0.557	0.848	0.325
TanhFix	**0.986**	0.019	0.943	0.023	0.951	0.029
TanhAdp	**0.990**	0.015	0.873	0.031	0.895	0.061
Gaussian	**0.976**	0.013	0.794	0.514	0.748	0.241
Beta	**0.970**	0.028	0.867	0.058	0.871	0.046

algorithm excels over the other meta-heuristic algorithms such as PSO and DE in the present experiment design with their respective parameter setting mentioned in Table I. However the algorithms PSO and DE have more performance tuning parameters than the ABC. Hence, their performance are subjected to meticulous tuning of their respective parameters. In contrast to van Wyk and Engelbrecht [9], we have performed experiments for the optimized parameters of Tangent-hyperbolic, Gaussian, and Beta function and the results suggests that performance of other adaptive transfer function are better than that of sigmoid function. Similarly, we have used ABC algorithms that performs better than that of PSO algorithm.

V. CONCLUSIONS

In present research, we have presented a meta-heuristic framework for the simultaneous optimization of the NN weights and the parameters of the transfer functions (NN-TFs model). Various types of transfer functions were chosen for the purpose of the rigorous analysis of the influence of the transfer functions optimization together with the NN weights. For the optimization of NN-TFs model, ABC, PSO and DE algorithm were used. Apart from the meta-heuristic algorithms, a Back-propagation algorithm was used for the comprehensive comparison and validation of the significance of the NN-TFs model. The comprehensive results presented in the paper suggests that the adaptive/customizable transfer function helps in enhancing the performance of NN. It may also be observed that the Beta function have four parameters and it is competitive to the Tangent-hyperbolic function that has two controlling parameters. Hence, further examination and tuning of its parameters may offer the best results in comparison to its counterparts. Apart from this, a probabilistic setting for the heterogeneity at the active nodes will be interesting to analyze. From the obtained results, it may also be observed that the ABC excels significantly over the other meta-heuristic such as particle swarm optimization in the present form of their parameter setting.

ACKNOWLEDGMENT

This work was supported by the IPROCOM Marie Curie initial training network, funded through the People Programme (Marie Curie Actions) of the European Union's Seventh Framework Programme FP7/2007-2013/ under REA grant agreement No. 316555.

REFERENCES

[1] S. Haykin, *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994.

[2] C. M. Bishop *et al.*, *Pattern recognition and machine learning*. springer New York, 2006, vol. 1.

[3] X. Yao, "Evolving artificial neural networks," *Proceedings of the IEEE*, vol. 87, no. 9, pp. 1423–1447, 1999.

[4] Y. Liu and X. Yao, "Evolutionary design of artificial neural networks with different nodes," in *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*, May 1996, pp. 670–675.

[5] D. White and P. Ligomenides, "Gannet: A genetic algorithm for optimizing topology and weights in neural network design," in *New Trends in Neural Computation*. Springer, 1993, pp. 322–327.

[6] I. Castelli and E. Trentin, "A preliminary study on training neural networks with adaptive activation functions."

[7] A. M. Alimi, R. Hassine, and M. Selmi, "Beta fuzzy logic systems: approximation properties in the mimo case," *International Journal of Applied Mathematics and Computer Science*, vol. 13, no. 2, pp. 225–238, 2003.

[8] H. Dhahri, A. M. Alimi, and A. Abraham, "Designing beta basis function neural network for optimization using artificial bee colony (abc)," in *Neural Networks (IJCNN), The 2012 International Joint Conference on*. IEEE, 2012, pp. 1–7.

[9] A. van Wyk and A. Engelbrecht, "Lambda-gamma learning with feed-forward neural networks using particle swarm optimization," in *Swarm Intelligence (SIS), 2011 IEEE Symposium on*, April 2011, pp. 1–8.

[10] W. Duch and N. Jankowski, "Survey of neural transfer functions," *Neural Computing Surveys*, vol. 2, no. 1, pp. 163–212, 1999.

[11] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Erciyes University, Engineering Faculty, Computer Engineering Department, Technical Report TR06, 2005.

[12] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (abc) algorithm," *Applied Soft Computing*, vol. 8, no. 1, pp. 687 – 697, 2008.

[13] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Micro Machine and Human Science, 1995. MHS '95., Proceedings of the Sixth International Symposium on*, 1995, pp. 39–43.

[14] A. Engelbrecht, *Computational Intelligence: An Introduction, 2nd Edition*. New York, NY, USA: Wiley, 2007.

[15] R. Storn and K. Price, "Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces," 1995.

[16] A. Slowik, "Application of an adaptive differential evolution algorithm with multiple trial vectors to artificial neural network training," *Industrial Electronics, IEEE Transactions on*, vol. 58, no. 8, pp. 3160–3167, 2011.

[17] E. Alba, *Parallel Metaheuristics: A New Class of Algorithms*. Wiley-Interscience, 2005.

[18] D. E. Rumelhart and J. L. McClelland, "Parallel distributed processing: explorations in the microstructure of cognition. volume 1. foundations," 1986.

[19] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 13, no. 2, pp. 398–417, 2009.

[20] D. Weingaertner, V. K. Tatai, R. R. Gudwin, and F. J. Von Zuben, "Hierarchical evolution of heterogeneous neural networks," in *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, vol. 2. IEEE, 2002, pp. 1775–1780.

[21] Z. Sheng, S. Xiuyu, and W. Wei, "An ann model of optimizing activation functions based on constructive algorithm and gp," in *Computer Application and System Modeling (ICCASM), 2010 International Conference on*, vol. 1, Oct 2010, pp. V1–420–V1–424.

[22] G. S. d. S. Gomes and T. B. Ludermir, "Optimization of the weights and asymmetric activation function family of neural network for time series forecasting," *Expert Systems with Applications*, vol. 40, no. 16, pp. 6438–6446, 2013.